CMPE 124 Lab 9: Flip flop circuits

Leonardo Blas

***Abstract*—The purpose of this lab project is to implement several flip flop digital circuits.**

I. Introduction

This lab project is based on the implementation of JK-type, T, and up-counter flipflops circuits. Furthermore, note that while we have already implemented counters in the past, we will now perform this task by cascading the memory stored by four 74_109 flip flops.

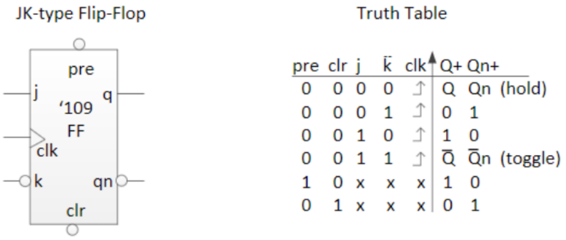
II. Design Methodology

For this lab project, we implement 4 flip flop circuits: a JK-type, two T, and an up-counter flipflops circuits. Each circuit depends on different input, and while some have different number of inputs than others, in the end they all result in the same output, q and not q. The JK-type flip flop will be implemented via a 74_109, the T flip flops via a 74_74 and 74_109 flip flops, and the up-counter via four 74_109s. Additionally, the implementation of our up-counter circuit involves the cascading of four JK-type flip flops. For further implementation details, see the latch circuits' diagrams and their truth tables in the upcoming figures.
Finally, to test our counter, we test it via a hex display.

A. Parts List

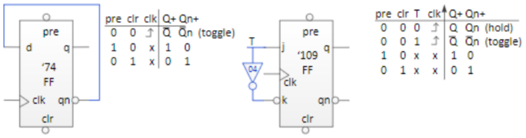
- Clock
- 5V source
- 74_109
- 74_74
- Binary witches
- Resistor
- NOT gates

B. Truth Tables



pre	clr	j	k̄	clk	Q+	Qn+	
0	0	0	0	↑	Q	Qn	(hold)
0	0	0	1	↑	0	1	
0	0	1	0	↑	1	0	
0	0	1	1	↑	$\overline{Q}$	$\overline{Q}$n	(toggle)
1	0	x	x	x	1	0	
0	1	x	x	x	0	1	

Figure 1. Truth table for our JK-type flip flop.



pre	clr	clk	Q+	Qn+	
0	0	↑	$\overline{Q}$	$\overline{Q}$n	(toggle)
1	0	x	1	0	
0	1	x	0	1	

pre	clr	T	clk	Q+	Qn+	
0	0	0	↑	Q	Qn	(hold)
0	0	1	↑	$\overline{Q}$	$\overline{Q}$n	(toggle)
1	0	x	x	1	0	
0	1	x	x	0	1	

Figure 2. Truth table for our SR latch using NAND gates circuit.

C. Schematics

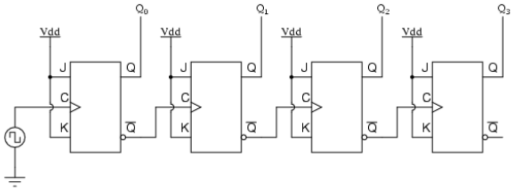


Figure 3. Up-counter circuit's form.

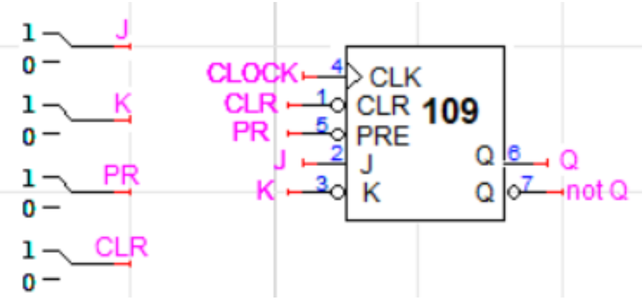


Figure 4. JK-type flip flop.

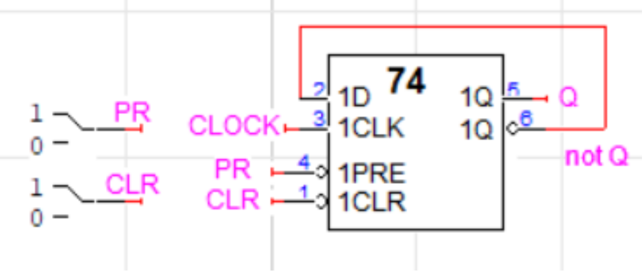


Figure 5. T flip flop implemented via a 74_74.

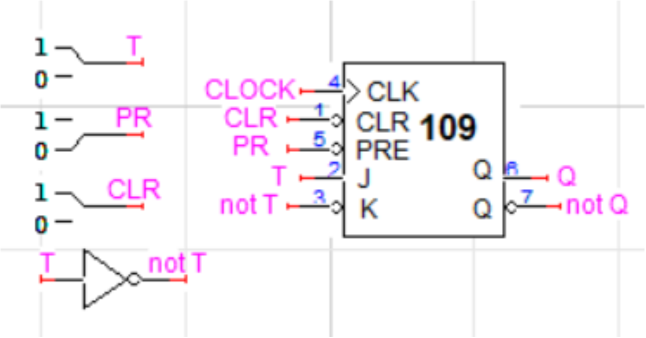


Figure 6. T flip flop implemented via a 74_109 component.

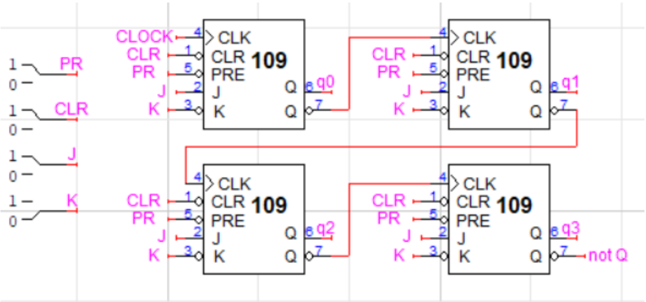


Figure 7. Up-counter circuit.

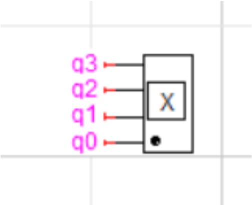


Figure 8. Hex display, to test our counter.

III. TESTING PROCEDURES

1. Create the four circuits, as portrayed in Figures 1 through 7.
2. Set up the binary switches to test each circuit's truth tables and the counter's functionality.
3. Reset the binary switches and run the simulation.
4. Record each circuits' waveform.
5. Verify that the hex display works as intended.

IV. TESTING RESULTS

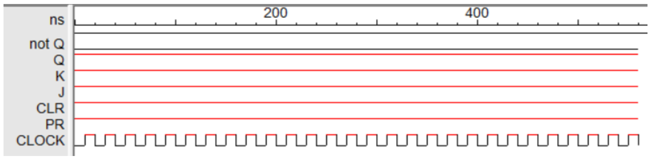


Figure 9. JK-type flip flop's sample waveform.

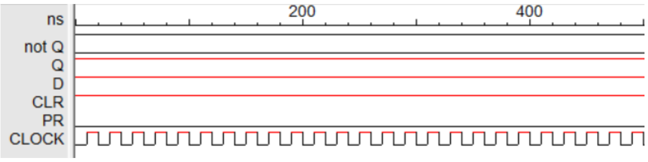


Figure 10. T flip flop implemented via a 74_74 's sample waveform.

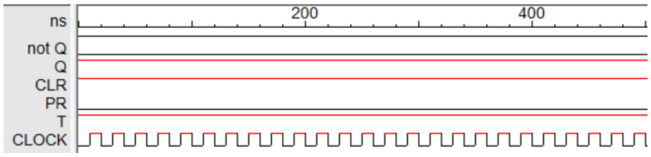


Figure 11. T flip flop implemented via a 74_109 's sample waveform.

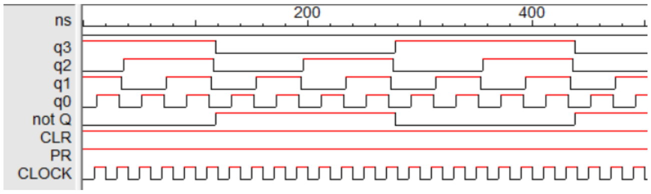


Figure 12. Up-counter circuit's sample waveform.

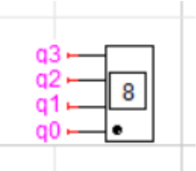


Figure 12. Hex display counting up, demonstrating that our up-counter circuit works appropriately.

V. CONCLUSION

In conclusion, there are several types of flip flop circuits that we can implement and experiment with, in order to further understand how a computer's memory works.
Additionally, through this lab we learned that not only we can use a flip flop's memory capacity to store binary numbers, but also to count up.
Furthermore, after checking our sample waveforms against their corresponding truth tables, as well as verifying our up-counter circuit via a hex display, we learned that we successfully implemented our latch and flip flops circuits.
Finally, this lab has taught us the basics behind the implementation of a computer's physical memory, which not only tests our theoretical knowledge—obtained in our lectures—on binary data, digital signals, and logic gates, but also advances our comprehension of logic circuits and how computers operate.
Ultimately, this lab taught us that flip flops and components with a memory aspect are not only useful to create computers, but all sorts of electronics, like a counter, in this case.